Modélisation Transactionnelle des Systèmes sur Puces en SystemC
Ensimag 3A — filière SLE
Grenoble-INP

TLM Avancé & Conclusion

Matthieu Moy
(transparents originaux de Jérôme Cornet)

Matthieu.Moy@imag.fr

2015-2016

Planning approximatif des séances

1. Introduction : les systèmes sur puce
2. Introduction : modélisation au niveau transactionnel (TLM)
3. Introduction au C++
4. Présentation de SystemC, éléments de base
5. Communications haut-niveau en SystemC
6. Modélisation TLM en SystemC
7. TP1 : Première plateforme SystemC/TLM
8. Utilisations des plateformes TLM
9. TP2 (1/2) : Utilisation de modules existants (affichage)
10. TP2 (2/2) : Utilisation de modules existants (affichage)
11. Notions Avancé en SystemC/TLM
12. TP3 (1/3) : Intégration du logiciel embarqué
13. TP3 (2/3) : Intégration du logiciel embarqué
14. TP3 (3/3) : Intégration du logiciel embarqué
15. 05/01: Intervenant extérieur : Jérôme Cornet
16. Perspectives et conclusion

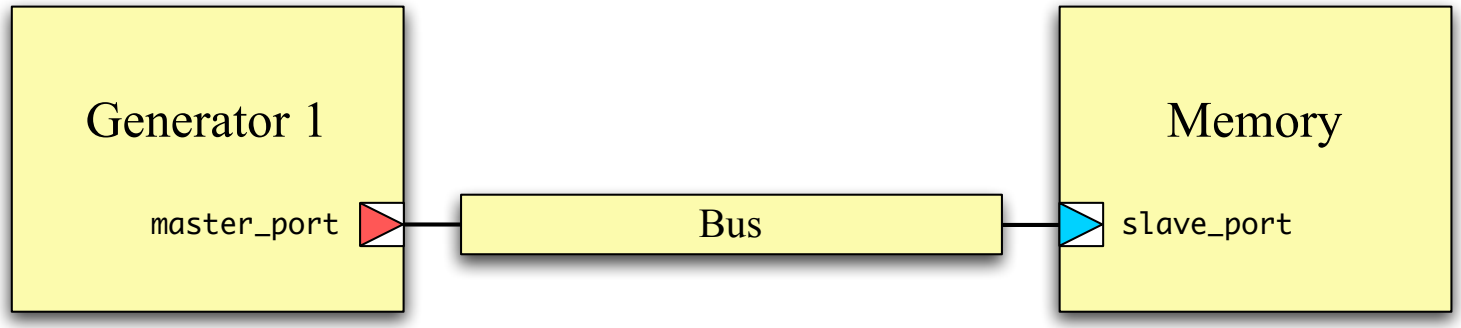
Grenoble INP ensimag

Sommaire

Sommaire

TP n°1

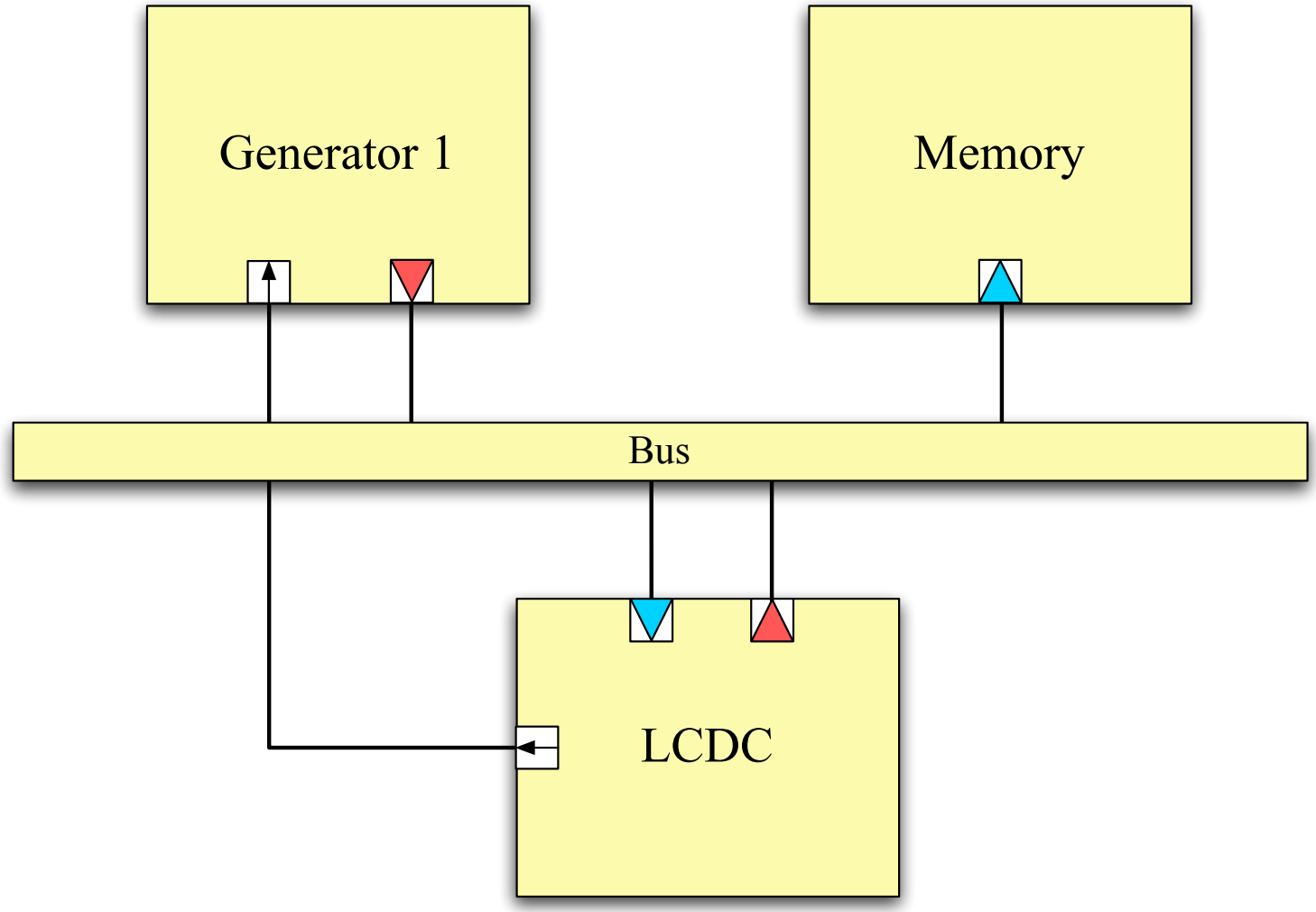
- Prise en main de SystemC/GCC
- Écriture d'un générateur de transactions
  - Outil de test de plateforme
  - Représente les accès que ferait un processeur (par ex)
- Écriture d'une mémoire
  - Mécanisme d'adresse locale (offset)
  - Implémentation du comportement (tableau dynamique C++)
- Comportement global



TP n°2

- Récupération des modules précédent
- Lecture de documentation technique : contrat d'utilisation du LCDC
- Modélisation de registres
  - Utilisation des événements SystemC
  - Correspondance avec la documentation
- Gestion des interruptions
- Fabrication d'images en mémoire...

TP n°2 - Figure



TP n°3

- Intégration du logiciel embarqué.
  - Avec ISS
  - En simulation native
- Correspondance entre plateforme physique (FPGA) et TLM
  - Même registres, même addressmap, même comportement
  - RAM programme gérée différemment
  - Protocole de bus non modélisé en TLM
- Logiciel portable via `hal.h`:
  - Une implémentation en simulation native
  - Une implémentation pour MicroBlaze (ISS ou FPGA)

TP n°3: Chaînes de compilation

- Native:
  - `g++`/`gcc`, comme d'habitude.
  - `extern "C"` pour faire communiquer le C et le C++ (problème de mangling et d'ABI)
  - Édition de liens entre plateforme et logiciel.
- Croisée:
  - `microblaze-uclinux-{gcc,ld,objdump}`: tourne sur x86_64, génère du code pour MicroBlaze.
  - Logiciel embarqué compilé en un fichier ELF ...
  - ... chargé dynamiquement en RAM par la plateforme.
  - `boot.s`: adresse de boot, vecteur d'interruption, ...
  - `it.s`: routine d'interruption (sauvegarde/restauration de registres avant d'appeler une fonction C)
  - `ldscript`: utilisé par `microblaze-uclinux-ld` pour décider des adresses des symboles.
  - `printf`: marche sur FPGA via une UART, trivial en simu native, composant UART en simu ISS.

TP n°3 : ce à quoi vous avez échappé...

- Fait pour vous:
  - Écriture des composants TLM (Giovanni Funchal)
  - ISS MicroBlaze, `boot.s`, `it.s` (SocLib)
- Non géré:
  - `gdb-server`: pour déboguer le logiciel avec `gdb` comme s'il tournait sur une machine physique distante.
  - Temps précis
  - Transaction bloc (entre RAM et VGA en particulier)
  - Conflits sur le bus entre RAM $\leftrightarrow$ VGA et fetch.
  - Contrôleur d'interruption évolué (le notre est essentiellement une porte « ou »)

Sommaire

Réutilisation de composants

- Point de vue d'un industriel:
  - Écriture de modèles TLM réutilisables de composants maisons
  - Modèles TLM de composants d'entreprises tierces?
- Idée : chaque fabricant de composant fournit plusieurs modèles
  - RTL ou netlist
  - Modèle TLM, etc.
  - etc.
- Problème : mettre tout le monde d'accord sur l'écriture de modèles TLM

Documentation

- Besoin d'informations organisées sur chaque composant
  - Banques de registres
  - Nombre de ports
  - Technologies de gravure supportées
  - Consommation électrique
  - Surface...

Documentation

- Besoin d'informations organisées sur chaque composant
  - Banques de registres
  - Nombre de ports
  - Technologies de gravure supportées
  - Consommation électrique
  - Surface...
- Création d'un consortium d'industriels pour standardiser les informations associées à un composant
  - Consortium SPIRIT : Structure for Packaging, Integrating and Re-using IP within Tool-flows
  - Standard IP-XACT.
  - Exemple de document : fichier XML conforme à un schéma
  - Création d'outils exploitant ces informations

Conclusion

- SystemC
  - « Langage » de modélisation niveau système
  - Utilisation par les industriels
  - Nombre conséquent d'outils
    - ★ Dédiés (CAD Vendors)
    - ★ Provenant de C++ (GCC, gdb, gprof, valgrind, etc.)
- TLM
  - Niveau émergent de modélisation de composants électroniques
  - Utilisation de SystemC
  - Existence d'outils spécifiques TLM (Cadence, Coware, Synopsys, ...)

Grenoble INP ensimag